Electronic Version
Stylesheet Version v1.1.1

Claims

[c1] 1. In a database system employing a transaction log, an improved method for restoring databases to a consistent version, the method comprising:
providing a shared cache storing database blocks for use by multiple databases;
for a read-only transaction of a given database, creating a cache view of the shared cache using the given database's transaction log, said cache view comprising particular database blocks of the shared cache that record a view of a particular version of the database at a given point in time;
creating a shadow cache for storing any database blocks that overflow said cache view; and
in conjunction with the cache view and the shadow cache, preserving a logical undo operation for the read-only transaction of the given database, so as to allow the given database to be restored to a transactionally consistent version upon starting the read-only transaction.

[c2] 2. The method of claim 1, wherein during occurrence of the read-only transaction any database blocks associated with the cache view are not written from the shared

cache back to disk.

[c3] 3. The method of claim 1, wherein the shadow cache is implemented via a temporary database table.

[c4] 4. The method of claim 1, wherein said logical undo operation allows the system to bring the cache view into a transactionally consistent state without having use a transactional clean point.

[c5] 5. The method of claim 1, wherein the shadow cache is used only if an overflow occurs.

[c6] 6. The method of claim 1, further comprising:
an allocation bitmap indicating database blocks in use.

[c7] 7. The method of claim 6, further comprising:
deleting the shadow cache simply by updating the allocation bitmap for allocated database blocks and then deleting the shadow table.

[c8] 8. The method of claim 1, wherein the shadow cache saves off database blocks that are computationally expensive to recreate.

[c9] 9. The method of claim 1, further comprising:
reusing the cache view in instances where database blocks of the cache view remained unchanged since starting the read-only transaction.

[c10] 10. The method of claim 1, further comprising:
upon termination of the read-only transaction, marking the cache view as closed.

[c11] 11. The method of claim 10, further comprising:
traversing the shared cache looking for database blocks to purge; and
reusing blocks from any cache view that has been marked as closed.

[c12] 12. The method of claim 1, further comprising:
reusing the cache view for other read-only transactions, if no new write operations have been committed.

[c13] 13. The method of claim 1, further comprising:
automatically detecting the read-only transaction; and
upon occurrence of write operations, posting back link log records that serve to link together blocks of the transaction log that pertain to the read-only transaction.

[c14] 14. The method of claim 13, further comprising:
if the read-only transaction must be undone, using the back link log records to skip portions of the transaction log that are irrelevant for undoing the read-only transaction, wherein the back link log records are only generated in the transaction log when there are active read only transactions.

[c15] 15. A computer-readable medium having processor-executable instructions for performing the method of claim 1.

[c16] 16. A downloadable set of processor-executable instructions for performing the method of claim 1.

[c17] 17. A database system capable of restoring databases to a consistent version, the system comprising:
a database system employing a transaction log;
a shared cache that stores database blocks for use by multiple databases;
a cache view of the shared cache created using the transaction log of a given database, said cache view being created in response to a read-only transaction of a given database, said cache view comprising particular database blocks of the shared cache that record a view of a particular version of the database at a given point in time;
a shadow cache for storing any database blocks that overflow said cache view; and
a module for preserving a logical undo operation for the read-only transaction of the given database, so as to allow the given database to be restored to a transactionally consistent version upon starting the read-only transaction.

[c18] 18. The system of claim 17, wherein during occurrence of the read-only transaction any database blocks associated with the cache view are not written from the shared cache back to disk.

[c19] 19. The system of claim 17, wherein the shadow cache is implemented via a temporary database table.

[c20] 20. The system of claim 17, wherein said logical undo operation allows the system to bring the cache view into a transactionally consistent state without having use a transactional clean point.

[c21] 21. The system of claim 17, wherein the shadow cache is used only if an overflow occurs.

[c22] 22. The system of claim 17, further comprising:
an allocation bitmap indicating database blocks in use.

[c23] 23. The system of claim 22, further comprising:
a module for deleting the shadow cache simply by updating the allocation bitmap for allocated database blocks and then deleting the shadow table.

[c24] 24. The system of claim 17, wherein the shadow cache saves off database blocks that are computationally expensive to recreate.

[c25] 25. The system of claim 17, further comprising:
a module for reusing the cache view in instances where database blocks of the cache view remained unchanged since starting the read-only transaction.

[c26] 26. The system of claim 17, further comprising:
a module for marking the cache view as closed, upon termination of the read-only transaction.

[c27] 27. The system of claim 26, further comprising:
a module for traversing the shared cache looking for database blocks to purge, and for reusing blocks from any cache view that has been marked as closed.

[c28] 28. The system of claim 17, further comprising:
a module for reusing the cache view for other read-only transactions, if no write operations have occurred.

[c29] 29. The system of claim 17, further comprising:
a module for automatically detecting the read-only transaction, and posting back link log records that serve to link together blocks of the transaction log that pertain to the read-only transaction.

[c30] 30. The system of claim 29, further comprising:
a module for using the back link log records to skip portions of the transaction log that are irrelevant for undoing the read-only transaction if the read-only transaction

must be undone, wherein the back link log records are only generated in the transaction log when there are active read only transactions.